NETWORK WORKING GROUP
Request for Comment #139
NIC #6717

T. C. O'Sullivan
Raytheon
May 7, 1971

Categories: D
RFCs Obsoleted - none
RFCs Updated - 137*

Discussion of TELNET Protocol

The attached discussion is an extension of RFC 137, NIC #6717, and is presented to provide useful background to designers and implementers to help them interpret the proposed Protocol and evaluate it in preparation for further discussion at the Atlantic City meetings.

While the views in the discussion represent those of various TELNET committee members, they should not be interpreted as being the agreed view of committee. They are the author's understanding of some of the arguments and background to the PROTOCOL proposed in the TELNET PROTOCOL recommendations.

---

* See Footnotes to attached discussion for changes to RFC 137.

Discussion of TELNET PROTOCOL

The use of a standard, network-wide, intermediate representation of terminal code between sites eliminates the need for using and serving sites to keep information about the characteristics of each other's terminals and terminal handling conventions, but only if the user, the using site, and the serving site assume certain responsibilities.

1. The serving site must specify how the intermediate code will be mapped by it into the terminal codes that are expected at that site.
2. The user must be familiar with that mapping.
3. The using site must provide some means for the user to enter all of the intermediate codes, and as a convenience, special control signals, as well as specify for the user how the signals from the serving site will be presented at the user terminal.

Other schemes were considered but rejected. For example, a proposal that the using site be responsible to transmit to and from the code expected by the serving site was rejected since it required that the using site keep tables of all serving site codes and provide mapping for each case. The information would require constant maintenance as new hosts were added to the network.

Since it is not known how the current or future sites will specify the mapping between the network-wide standard code (7 bit ASCII in an 8 bit field) and the codes expected from their own terminals, it seems necessary to permit the user to cause every one of the 128 ASCII codes, plus (for full user power) selected control signals (either of a TELNET control nature, or of a special terminal nature such as break or attention).

There was strong feeling about the importance of the user/system interface at the using site, but equally strong feeling that this problem is one of local implementation and should reflect the using site installation philosophy rather than the subject to network-wide standards. Some topics of consideration in this area are:

1. How to represent special graphics, not available at the using site, at the user's terminal.
2. Treatment of upper/lower case problem on TTY 33 and 35.
    a. Representing lower-case output.
    b. Providing users with shift and shift lock signals.
3. Incorporating editing capability in TELNET.
4. Extending user options in Network mode not available to local users, e.g., hold output
    kill print
5. Permit users to specify how keyboard input is to be translated, e.g., let a character from the terminal cause a specified string to be sent by the user's TELNET.

In early discussions, there was pressure to get a simple statement of protocol out early to permit early use of selected systems. The counter pressure to provide a richer set of protocol in the first release was also present. Work started in the direction of the latter, but the complexities introduced were not necessary for early use of the network. The proposed solution to the TELNET protocol problem seems to provide a mechanism for a minimum implementation (to be discussed later) while providing a basis for developing richer sets of protocol for present and future use in terminal applications, process-process communications, and use by other conventions to pass data or control information.

The understanding that ASCII be used as a network-wide code has been established for some time. Its use in TELNET provided a problem with respect to the limitation of a maximum character set of 128. Some systems provide for more than this number in their operation, and therefor, as serving sites cannot map on a one for one basis.

Each such serving site could probably provide a reasonably useful character set, including all system control signals, by mapping 128 of its codes and just not provide a network user access to the other codes. However, any character left out might later be used in a major application at that site as a special control signal. This could result in denying network users the facility offered by that application. Serving sites are, therefor, encouraged to provide a full mapping between the ASCII code and the code used on the serving system.

The ASCII code for ESC (known to some as ALT MODE) has been selected as an escape*. For each serving site character not mapped on a one for one basis, the serving site can specify an escape character or string of escape characters (preferably a printable graphic) to represent it. Thus, the user could enter the full set of serving site code from any network terminal operating through the Network Virtual Terminal (NVT) ASCII convention. The serving site, in generating output directed at the user's terminal, would be expected to map out such a character and transmit the appropriate ESC character or string of ESC characters.

> Example: A serving site, whose normal code is EBCDIC, has specified that cent (¢) has not been mapped on a one for one basis and that to transmit the character, users must enter ESC followed by C. At a using site, the TELNET implementers have decided to try to print out all ESC characters using \ to indicate ESC. On receipt of the representation for cent, the user would see \ C on his print-out.

The representation of the end of a physical line at a terminal is implemented differently on network HOSTS. For example, some use a return (or new line) key, the terminal hardware both returns the carriage or printer to start of line and feeds the paper to the next line. In other implementations, the user hits carriage return and the hardware returns carriage while the software returns to the terminal a line feed. The network-wide representation will be carriage return followed by line feed. It represents the physical formatting that is being attempted, and is to be interpreted and appropriately translated by both using site and serving site.

> Example: A Multics user is working, through the network, on some serving site HOST. In the course of the session, the user has numerous occasions to hit New Line on his Mod 37 TTY. Each time the Multics system is awakened by a New Line interrupt, the line of buffered characters is passed to TELNET where it is scanned for special characters. If none is found, carriage return followed by line feed is inserted where New Line was entered, and the line is turned over to the NCP for transmission. When the TELNET finds the carriage return line feed sequence in the data stream coming from the serving site, the two characters are replaced with New Line code and sent to the terminal.

---

*Please drop the (s) at the end of "character" in paragraph 3, page 3, RFC 137, NIC #6714.

The decision to have the assumed condition for echo be that the using site will provide any echo necessary for its terminals was taken because of the difficulties faced by some installations that cannot turn off their echo or that have terminals that print locally as a result of key strokes. Serving sites could take the position "let the user turn my echo off", but this seems an unnecessary burden on the user. In addition, some serving sites may choose not to supply any echo service, in which case the no echo assumption will supply a network-wide condition, while other assumptions would give a mixed starting condition.**

The convention of using "I ECHO", "YOU ECHO" seems to fill both the requirements for dynamic echo control and for a minimum implementation of TELNET Protocol*. An agreed-upon exchange to pass echo control (i.e., two sites exchange the I ECHO/YOU ECHO codes) results in passing the control from one site to the other.

Example: A serving site is exchanging control information with the USER in an area where the serving system asks for pass word and wants to suppress the printing of the pass word at the using site's user terminal. (In this case, the using site has the ability to control the print capability at the user's terminal.) Using site has been echoing to the user's terminal.

Serving Site to Using Site (→)
I ECHO
Using Site to Serving Site (←)
YOU ECHO
→Pass word:
←(User enters password at terminal)
→(No echo sent)
→YOU ECHO
←I ECHO

---

*Note: Please change RFC #137, NIC #6714, page 4 - Code X'85' to read Reserved.

**Also make note that the starting assumption in the initial exchange between using site and serving site will be that the using site will (if necessary) provide echo and the serving site will not.

After the exchange, the original normal condition is re-established. If the using site did not have dynamic echo control installed in its TELNET implementation, the serving site would have signaled I ECHO several times, received no response, and assumed that the using site could not comply proceeding to call for the pass word without the normal protection of inhibiting print.

TELNET control signals are of two types: one that results in transmission of signals down the network to a receiving site; the other intended for the user/process site only. The latter type will be discussed later. So far, we have discussed the former type, specifically dealing with echo control.

The use of ESC should not be considered a TELNET-wide standard, but a convention limited to the 7 bit ASCII mode of transmissions. Other conventions, to be incorporated later, may include binary transmission, EBCDIC, etc. Presumably, each will have its own convention for an escape character to extend its code set.

Since it is expected that conventions other than ASCII will be implemented under TELNET, a code to indicate a DATA TYPE representing each set of conventions will be employed. The control code X'A0' has been selected to represent the ASCII convention in TELNET. Since a number of applications may wish to transmit transparently (i.e., 8 bit binary data), X'A1' is being reserved for that purpose. The TELNET control code X'A2' is reserved for an expected set of EBCDIC conventions. The DATA TYPE is expected as the first byte of data over a TELNET connection. Minimum implementations will be aided by providing a default. That is, if the first byte over a connection has the high order bit set to zero, then the transmission has begun in ASCII mode.

Each set of conventions, i.e., each DATA TYPE will be expected to have a convention for that DATA TYPE to signal that it is returning to control mode. This return may be for the purpose of making use of an existing control codes or to change data type. X'88' is used*.

---

*Please note on page 4 of RFC 137 that the receipt of an X'88' should be responded with by the receiver sending a double signal, i.e., X'88'X'88' if the new DATA TYPE can be handled.

Example: At the using site, a terminal has a special device on it (e.g., plotter, laboratory instrument, control box, etc.) that is controlled by binary code in 8 bit bytes. The terminal uses a special "enter" code that routes signals to the device and cuts off printing at the terminal until a special "leave" signal is received from the driving process. The driving process in this case is at a remote serving site. It is assumed in this example that a DLE convention is used for transparent transmission, a single DLE signal representing return to control. Normal transmission has been in ASCII.

Driving Process (at Serving Site) to Using Site) →

X'88'X'A1'

Using Site to Serving Site ←

X'88'X'88'

→

ENTER code...8 bit binary bytes...

Using Site TELNET to Terminal ↓

ENTER code...8 bit binary bytes...

Terminal

Turn printer off, feed transparently to special device, look for LEAVE signal

→

8 bit binary bytes...LEAVE signal...single DLE

X'A0'

←

X'88'X'88'

→

Message

↓

8 bit binary data...LEAVE signal MESSAGE

Terminal

During this sequence of exchanges - at the terminal, feed binary data to special device until LEAVE signal is sensed, strip off LEAVE signal, turn on printer and block data path to special device, print MESSAGE at terminal.

There is a special control signal on some terminals that has no corresponding bit pattern in ASCII, but is transmitted by a special electrical signal. This control signal is ATTN on a 2741 and BREAK on a teletype. The ASCII DATA TYPE in TELNET will use the code X'81' to represent BREAK. (There is a corresponding control signal for use from serving sites to using sites for reverse break, and it is assigned the code X'82'.)

Some systems treat the break as an extra code available for use in conjunction with the data stream. For example, one system uses break as a special editing code meaning "delete the current line to this point". In these cases, the code may simply be inserted in the data stream with no special additional action by the user.

Other systems use BREAK or ATTN in a special interrupt fashion, to mean stop processing the application and give me the supervisor, or cancel the present job, etc. (Other systems use normal characters for this purpose, such as "Control C".) In these cases, because of differences in the ways both serving and using sites operate, it is necessary to take a route in addition to the normal TELNET data stream to signal that the special control signal is imbedded in the data stream.

Examples - Problem

The PDP-10 normally will, when it fills its input buffer, continue to accept characters from a terminal examining each to see if it is a control character, then act on it if it is or throw it away if it is not.

Since the TELNET server at the serving site is at the mercy of the NCP with respect to controlling the bunching, and therefor, arrival at the TELNET of bursts of characters, TELNET implementations might be expected to choke off flow to the buffers until they are ready to accept characters without throwing them away.

Under this condition, the serving process might be outputting to the using terminal, the input buffers fill up, and a control C get stuck in the data stream that has been choked off.

A similar problem could occur with the Multics or some IBM system as a server. The user at a using site gets into an output loop at the serving site and wants to break the process without having to release his TELNET connection. The buffers clog the connection, transmission is choked off, and the control C, break, or other user control signal gets stuck in the pipeline.

Example - Solution

The user at the using site knows he is entering a special control signal (break, ATTN, control C, etc.) and follows it with an X'80'. (The local instructions at using sites for accomplishing this may differ from site to site.)

Using Site TELNET to Serving Site
    Insert X'80' in Data Stream
Using Site TELNET to Using Site NCP
    Send an INS
Serving Site NCP to TELNET Server
    Look out, here she comes
Serving Site TELNET
    Does its special thing until it sees X'80' then resumes normal handling

Thus, depending on the server's local implementation to provide adequate service, a special handling of the data stream can be invoked whenever an INS is received in order to get to the special character. When it sees X'80', it recognizes it as a SYNC character and knowning that the special character has been passed on, strips the X'80' from the data stream and returns to normal mode.

If the X'80' arrives before the INS, a counting scheme can keep the activity appropriate to the serving site conditions.

This approach to handling selected special characters or signals relieves the using TELNET processes from having to recognize the special serving site characters, as well as from having to know how the serving site wants to handle them. At the same time, the procedure requires only a minimum level of user understanding of the serving site. This seems appropriate, since the TELNET ASCII conventions are providing a Network Virtual Terminal, not a Network Virtual User.

The ability of the user to cause the using site TELNET to send any combination of ASCII characters in a string, and only that combination, is viewed as important to the user utility of the TELNET ASCII conventions. Because of this, some user sites may find it necessary to provide special local TELNET control signalling from the user to the using site TELNET.

Examples

A user on a line at a time system (Multics, System 360, GECOS, etc.) is working through the Network on aserving site that operates a character at a time. The application is a debugging aid that permits the user to type in a memory location = to which it will respond with n where n represents the current contents of that location. The serving site process does not expect to see the location = followed by a carriage return line feed sequence. The user at the using site should be able to type in the location, follow it with a signal to suppress the end of line convention, followed by new line or return, and expect the location number = to be transmitted immediately without an end of line sequence.

In another case, a using site has decided that it is convenient to accumulate four characters at a time and transmit them to the serving site, unless an end of line is observed, in which case the end of line sequence is sent preceded by whatever number of characters have been accumulated, (presumably three or less). In the same debugging application, the address is such that the end does not correspond with the four character buffer demarcation. The user should have the ability to enter a code for "transmit immediately" in place of the Carriage Return in order to preserve neat formatting, and expect the address to be sent to the serving site.

TELNET controls have been discussed and those introduced to date are probably sufficient for an early implementation of TELNET ASCII convention. There will be a need to establish a mechanism for the controlled assignment (on request by Network Sites), and announcement of DATA TYPE and CONTROL codes.

It should be noted that some controls are network-wide TELNET controls, while others are specific to the ASCII Data Type. It should be further recognized that some local control messages do not require a corresponding network-wide code.

While it is recognized that even a minimum implementation of TELNET for a using site is expected to permit the user to send any selected ASCII string (and only that string) to the serving site, it is not necessary for a serving site to implement a full mapping from ASCII to local code, nor is it necessary for either the using or serving sites to implement all control codes.

Example - Using Site

A minimum implementation of the TELNET protocol for the using site would permit ignoring (and stripping) any control signals from the serving site since they would all either require agreement or acknowledgement (e.g., DATA TYPE, ECHO CONTROL, etc.) or can be ignored with no particularly harmful results (e.g., reverse break).

Example - Serving Site

A minimum implementation of the TELNET protocol for the serving site could provide one for one mapping for the most important 128 serving system controls and graphic signals, and ignore all control signals.

It would be helpful if a minimally implemented receiving site, when it recognizes an incoming control signal for which appropriate reaction is not available, could respond with X'87' (The following not implemented at this site) and follow it with the code just received.

Whenever an ASCII TELNET connection is lost, it should be assumed that the process at the other end of the connection has been quit, aborted, failed, etc. In this way, a minimum using site installation can fail to implement the break and break synchronization, and have the user rely on the using site local procedure for leaving a running local process and returning to the supervisor to break a connection to a remote serving site.

Example

User recognizes that he is caught in an output loop and wishes to stop his user process at the serving site. The serving site requires a break, but the using site minimum implementation has not made it available. Even if it had, the INS was not implemented and could not be used to unblock the input pipe. Locally, the using site convention for leaving a process and getting to supervisory level is to hit the attention key on the 2741 terminal. The user does this and is passed to the supervisor where he signals to release the TELNET connection. The serving site, seeing that an ASCII TELNET connection has been lost, assumes that the user is ended either normally or abnormally. Serving site cancels the user's process. The user tries again by re-establishing the connection, logging in again, re-initiating the process, etc.

Other conventions under TELNET may make quite different assumptions about lost connections, and some may go as far as dynamic establishing and releasing of connections.

The proposed TELNET ASCII implementation leaves much uncovered, but seems to permit early simple implementation with varying levels of capability, along with the capacity to expand in several ways to meet other needs.

There is an important open question. Should a PROTOCOL such as TELNET provide the basis for extending a system to perform functions that go beyond the normal capability of the local system. For example, a local system may not provide functions such as Hold Output, Kill Print, etc., but could extend it for network purposes through TELNET. If so, to what extent should such extensions be thought of as Network-wide standards as opposed to purely local implementations.